附件1：实验报告中的诚信设计

数据结构与算法导论

实验报告

实验题目： 二叉排序树实现

姓 名： 梁睿鸣

学 号： 2020210139

日 期： 2021/6/17

自我评分： A+ 【 X 】 95

自我评分说明：A+，A，B+，B，B-，C，D，分别对应分数95、90、85、80、75、70、60

诚信声明

本人郑重承诺：本实验程序和实验报告均是本人独立学习和工作所获得的成果。尽我所知，实验报告中除特别标注的地方外，不包含其他同学已经发表或撰写过的成果；实验程序中对代码工作的任何帮助者所作的贡献均做了明确的说明，并表达了谢意。

如有抄袭，本人原因承担因此而造成的任何后果。

特此声明。

签名：梁睿鸣

日期：2021/6/17

程序引用说明

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 引用项 | 来源 | 代码引用行数 |
| 1 | 删除函数 | 课本 | 30 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| 小计 1 | | | 30 |

总代码行数 166 ； 引用占比 18.1%

1、实验简介

【实验内容的简要说明，具体说明实验完成的功能和性能要求】

根据栈的抽象数据类型的定义，按要求实现一个链栈

并编写测试main()函数测试栈的正确性。

2、程序框架

public:

    BST(T r[],int n); //构造函数

    ~BST(); //析构函数

    BiNode<T> \*Search(T key); //公有搜索函数

    void Insert(T key); //公有插入函数

    bool DeleteBST(T key); //公有删除函数

    void PrintBST(); //打印函数

private:

    BiNode<T> \*Search(BiNode<T> \*&R, T key); //私有搜索函数

    void Insert(BiNode<T> \*&R, BiNode<T> \*s); //私有插入

    void Delete(BiNode<T> \* &R); //私有删除节点

    bool DeleteBST(BiNode<T> \*&R, T key); //私有删除

    void ReleaseNode(BiNode<T> \*&R); //私有删除节点元素

3、关键代码实现

【哪些函数是你认为最能体现自己工作成果的函数，说明函数实现基本思想（可用文字或图表示），以及具体的实验步骤（用伪代码或带注释代码）】

3.1 ReleaseNode(BiNode<T> \*&R)

基本思想：

如果该节点为空则退出

先递归删除左子树

再递归删除右子树

删除自己

template<class T>

void BST<T>::ReleaseNode(BiNode<T> \*&R)

{

    if(R == NULL)

        return;

    ReleaseNode(R->lch);

    ReleaseNode(R->rch);

    // cout<<"Releasing Node:"<<R->data<<endl;

    delete R;

}

3.2 PrintBST(BiNode<T> \*&R)

基本思想：

按照中序遍历输出树

void BST<T>::PrintBST(BiNode<T> \*&R)

{

    if(R == NULL)

        return;

    PrintBST(R->lch);

    cout<<R->data<<endl;

    PrintBST(R->rch);

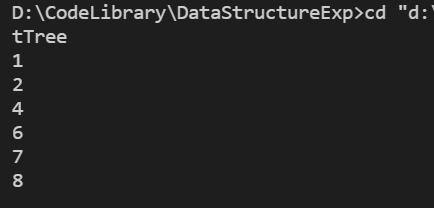
}

4、不足

【实验程序哪些函数功能还有缺陷或不足，或者程序架构有不足，或者性能还有待提高、或者代码不和规范等等，所有你自己对程序不满意的地方】

函数的传值可以改进，可以不需要传入初始指针，直接从根节点开始运行

5、运行效果图



6、心得体会

【所有你在实验中的感受和想和老师说的话都可以放在这里，篇幅不限】

书本的代码写得很巧妙，自己的实现稍显冗杂，可以多学习课本的思路尝试提高代码的效率。